CSE598i - Web 2.0 Security

Introduction

Professor Patrick McDaniel
Spring 2010
A bedtime story ...
This course

- We are going to explore why this event is not *isolated*, *infrequent*, or even *unexpected*.
- Why are we doing so poorly in web systems at protecting our users and data from inadvertent or intentional harm?

The answer: stay tuned!
This course ...

• This is a development and reading course covering topics in web systems design and security, including:
  ‣ web design, AJAX, cloud computing web vulnerabilities, web design, static analysis ....
You need to understand ...

- IP Networks
- Operating Systems
- System administration
- Basic web architecture
- Coding
- XML
Goals

‣ My goal: to provide you with the tools to understand and develop secure web 2.0 systems.

‣ Basic technologies
‣ Engineering/research trade-offs
‣ How to read/write/present security research papers

• This is going to be a low impact course. However, the key to success is sustained effort. Failure to keep up with readings and project will likely result in poor grades, and ultimately little understanding of the course material.

• Pay-off: security in web systems is highly sought skill
Course Materials

• Website - I am maintaining the course website at
  ‣ http://www.cse.psu.edu/~mcdaniel/cse598i-s10/

• Course assignments, slides, and other artifacts will be made available on the course website.

• No Course textbook, but books on CSS, XML JQuery, or Apache may be helpful (but there are plenty of online resources)
Course Calendar

• The course calendar as all the relevant readings, assignment due dates

• The calendar page contains electronic links to online papers assigned for course readings.

• **Please check the website frequently for announcements and changes to the schedule.** Students are responsible for any change on the schedule.
Grading

• The course will be graded on projects, paper summaries, paper presentations, and class participation in the following proportions:

  40% Course Project
  30% Course Lectures
  20% Paper Summaries
  10% Class Participation
Course Project

• Build a Web 2.0 system from the ground up.
• This is a development/engineering exercise, not a research exercise (however, you will gain skills that may support later research).
• You decide what to build, but it must, at a minimum:
  ‣ Support users/logins
  ‣ Use one of the Google APIs
  ‣ Scrape data from external website
  ‣ Export API to other projects in the class
  ‣ Import API from other projects in the class
• Hint: build something you really want.
Lectures

• Course papers will be presented by students. Student presenters identified on the webpage will create a 20 minute lecture of original content.

• Each lecture should roughly follow this outline:
  1. Problem area/motivating example
  2. Background literature
  3. Key concepts and terminology
  4. Solution
  5. Evaluation/Validation
  6. Take aways
Example: PSUSportsWeb

- PSUSportsWeb is an application that provides the user to track all of the sports from Penn State, and to exchange information with other users.
  - Users sign up (*support users/logins*)
  - See a annotated map of all sports teams next events (*use one of the Google APIs*)
  - Live update of all sports teams upcoming events, scores, and statistics (*scrape data from external website*)
  - Provide XML feed of scores (*export API to other projects in the class*)
  - Extract temperature at events from weather application (*import API from other projects in the class*)
Students will complete a summary for each paper assigned for class.

- Templates downloaded from course website (schedule)
- The XML will be filled out (instructor, student)
- Each XML will be placed on the website completed as part of assignment #1.

<SUMMARY>


<OVERVIEW>
Your 5-6 sentence overview of the paper. Close to an abstract in your own words.
</OVERVIEW>

<APPROACH_RESULT>
This should be a 5-6 summary of what the authors where attempting find how, how, and what they found.
</APPROACH_RESULT>

<QUESTIONS>

<QUESTION>
<QUERY>Which of the vulnerabilities listed in the paper could be considered as remotely controlling the browser? (explain why for each)</QUERY>
<ANSWER>2-3 sentence answer to the question.</ANSWER>
</QUESTION>

<QUESTION ...>
</QUESTIONS>
</SUMMARY>
Ethics Statement

• This course considers topics involving personal and public privacy and security. As part of this investigation we will cover technologies whose abuse may infringe on the rights of others. As an instructor, I rely on the ethical use of these technologies. Unethical use may include circumvention of existing security or privacy measurements for any purpose, or the dissemination, promotion, or exploitation of vulnerabilities of these services. Exceptions to these guidelines may occur in the process of reporting vulnerabilities through public and authoritative channels. Any activity outside the letter or spirit of these guidelines will be reported to the proper authorities and may result in dismissal from the class and or institution.

• When in doubt, please contact the instructor for advice. Do not undertake any action which could be perceived as technology misuse anywhere and/or under any circumstances unless you have received explicit permission from Professor McDaniel.
What is the web?

• A collection of application-layer services used to distribute content
  ‣ Web content (HTML)
  ‣ Multimedia
  ‣ Email
  ‣ Instant messaging

• Many *browser driven* applications
  ‣ News outlets, entertainment, education, research and technology, …
  ‣ Commercial, consumer and B2B
  ‣ Social networking
Network vs. Web Security
Web security challenges

- The largest distributed system in existence
  - threats are as diverse as applications and users ...
  - built on a culture of hack and deploy ... (*rapid prototyping*)

- The stakeholders are ...
  - consumers (users, businesses, *agents*, …)
  - providers (web-servers, IM services, …)

- Another way of seeing web security is
  - Securing the web *infrastructure* such that the *integrity*, *confidentiality*, and *availability* of content and user information is maintained
A brief history of web technology ...
Early Web Systems

- Early web systems provided a click-render-click cycle of acquiring web content.
  - Web content consisted of static content with little user interaction.
Adding State to the Web: Cookies

- Cookies were designed to offload server state to browsers
  - Not initially part of web tools (Netscape)
  - Allows users to have cohesive experience
  - E.g., flow from page to page,
- Someone made a design choice
  - Use cookies to authenticate and authorize users
  - E.g. Amazon.com shopping cart, WSJ.com
Web Transport Security: SSL

- Secure socket Layer (SSL/TLS)
- Used to authenticate servers
  - Uses certificates, “root” CAs
- Can authenticate clients
- Inclusive security protocol
- Security at the socket layer
  - Transport Layer Security (TLS)
  - Provides
    - authentication
    - confidentiality
    - integrity
Dynamic Content: CGI

- Common Gateway Interface (CGI)
  - Generic way to call external applications on the server
  - Passes URL to external program (e.g., form)
  - Result is captured and return to requestor

- Historically
  - “shell” scripts used to generate content
    - Very, very dangerous

- NOTE: server extensions are no better (e.g., servlets)
Applications/Plugins

• A **plugin** is a simply a program used by a browser to process content
  ‣ MIME type maps content to plugin
  ‣ Like any old application (e.g., RealAudio)
  ‣ Newer browsers have autoinstall features

• A kind of plug-in …
  ‣ (1997) David.exe
  ‣ “Free pornography …”

• Moral: beware of plugins
Java

• Platform and language for writing applets
  ‣ Sun Microsystems platform for set-top boxes
  ‣ Applets embedded in web pages (or native)
  ‣ Language loosely resembling C++
  ‣ Runs in a Java Virtual Machine (JVM)
    • Every platform has JVM
    • Platform runs arbitrary code (bytecode)
    • Hence: one application runs on a bunch of platforms
    • Great way to take advantage of the web
    • Slow for data/processing intensive applications
Active X

• ActiveX is a MS windows technology
  ‣ Really, just a way to run arbitrary code
  ‣ Called controls (.OCX), just programs
  ‣ Conforms to MS APIs to interact with web
• Extends user experience in lots of nice ways
  ‣ Microsoft upgrade service
  ‣ BIOS Upgrades
  ‣ Lookup services
• Massive security hole ....
Authenticode/Signed .jar(s)

- Problem: I need to run an application code on my machine, but I worry about security
- Solution: Make sure code only comes from people that you trust.
- Authenticode
  - Sign download content
  - Check that signer is “trusted”
  - Used for all Win* content
- Problem: Jan 2001
  - Verisign issued two bad MS
JavaScript

- Scripting Language used to improve the quality/experience
  - Create dialogs, forms, graphs, …
  - Built upon API functions (lots of different flavors)
  - No ability to read local files, open connections …

- Security: No ability to read local files, open connections, but …
  - DOS – the “infinite popup” script
    - Often could not “break out” with restarting computer
  - Spoofing – easy to create “password” dialogs
DC: Embedded Scripting

• Program placed directly in content, run on server upon request and output returned in content
  ‣ MS active server pages (ASP)
  ‣ PHP
  ‣ mod_perl
  ‣ server-side JavaScript
  ‣ python, ....

• Nice at generating output
  ‣ Dangerous if tied to user input
The web has evolved from a *document retrieval* and rendering to sophisticated *distributed application platform* providing:

- dynamic content
- user-driven content
- interactive interfaces
- multi-site content content
- ....

With new interfaces comes new vulnerabilities ...
The new web-page

- Rendered elements from many sources containing *scripts*, *images*, and stylized by *cascading style sheets* (CSS)

- A browser may be compromised by any of these
Web-server APIs

- Web-servers often provide application extension APIs to which developers can build ...
  - ISSAPI
  - Apache API

- Act as kinds of “kernel modules” for web-server
  - Web-server processes received inputs (URL, fields, etc.)
  - Passes result to custom code (typically, C code)
Application Frameworks

- Application frameworks are software stacks that implement web application
  - Programmer adds domain-specific programming
  - Handle request handling and rendering
  - Quickly implement web apps without dealing the the nasty details of HTTP/HTML
- For example, the Zend framework implements a web application by processing incoming URLs
  - E.g., http://base/module/function
  - Zend accepts returned framework objects and renders them via internal API
  - Modify documents on the fly using AJAX scripts such as JavaScript
• **AJAX: asynchronous JavaScript and XML**
  - A collection of approaches to implementing web applications
  - Changes the click-render-click web interface to allow webpages to be interactive, change, etc.
  - Examples: Google Gmail/Calendar, Facebook, ...
  - Hidden requests that replace document elements (DOM)
Where we go from here.

• We will spend the semester looking into the security problems introduced by these technologies, with a focus on Web 2.0 systems ...
Assignment #1

• Part 1: install/configure the VM
  ‣ Configure the firewall and install and configure the Apache web server on VM.
  ‣ The firewall should not allow any traffic coming from anywhere outside PSU, and restrict ssh traffic to 130.203.22.184 and 130.203.32.241 and a single login host you wish to use. Port 80 should be open to the entire Internet.
Assignment #2

• Part 1: create summary page
  ‣ Use JQuery UI Widgets to design a summary webpage that:
    • Organize course lectures into "accordions" titled by topic
    • Papers within accordion are organized as tabs.
  ‣ Each summary is placed into an XML document downloaded/rendered when accordion is maximized.
  ‣ Each paper will have its own XML document, whose template can be retrieved from the course schedule page.
  ‣ Students should feel free to make the page look as nice as possible by adding any features or design.

• Note this **must** be done by next wednesday to perform the in class exercise.